**Lab 6: Dijkstra’s Shortest Path Algorithm**

**OBJECTIVE:**

1. To understand Dijkstra’s algorithm,
2. To write a C program to implement Dijkstra’s algorithm.

**THEORY:**

Dijkstra’s algorithm is an algorithm for finding the shortest paths between nodes in a weighted graph. This algorithm works for directed and undirected graphs with non-negative weights. It is a greedy algorithm.

**PSEUDOCODE:**

**procedure** *Dijkstra*(G: weighted connected simple graph, with all weights positive)

{G has vertices a = *v*0,*v*1,...,*v*n = z and lengths *w*(*v*i,*v*j) where *w*(*v*i,*v*j) = ∞ if {*v*i,*v*j} is not an edge in G}

**for** i := 1 **to** n L(*v*i ) := ∞

L(a) := 0 S := ∅ {the labels are now initialized so that the label of a is 0 and all

other labels are ∞, and S is the empty set} **while** z ∉ S

u := a vertex not in S with L(u) minimal S := S ∪ {u} **for** all vertices *v* not in S

**if** L(u) + *w*(u, *v*) < L(*v*) **then** L(*v*) := L(u) + *w*(u, *v*) {this adds a vertex to S with minimal label and updates the labels of vertices not in S}

**return** L(z) {L(z) = length of a shortest path from a to z}

**DISCUSSION:**

Dijkstra’s algorithm was implemented in the C programming language and the output was seen for a graph. The program correctly calculated the shortest path.

**CONCLUSION:**

This practical work gave me a hands-on experience which solidified my understand of Dijkstra’s algorithm.

**Lab 7: Kruskal’s Minimum Spanning Tree**

**OBJECTIVE:**

1. To understand Kruskal’s algorithm,
2. To write a C program to implement Kruskal’s algorithm.

**THEORY:**

A minimum spanning tree of a connected weighted graph is a connected subgraph, without cycles, for which the sum of the weights of all the edges in the subgraph is minimal. Kruskal’s algorithm finds the minimum spanning tree of an undirected weighted graph. It is a greedy algorithm which in each step add to the tree the lowest-weighted edge that will not form a cycle.

**PSEUDOCODE:**

**procedure** *Kruskal*(G: weighted connected undirected graph with n vertices) T := empty graph **for** i := 1 **to** n − 1

e := any edge in G with smallest weight that does not form a simple circuit when added to T

T := T with e added **return** T {T is a minimum spanning tree of G}

**DISCUSSION:**

Kruskal’s algorithm was implemented in the C programming language and the output was seen for a graph. The program correctly calculated the minimum spanning tree.

**CONCLUSION:**

This practical work gave me a hands-on experience which solidified my understand of Kruskal’s algorithm.

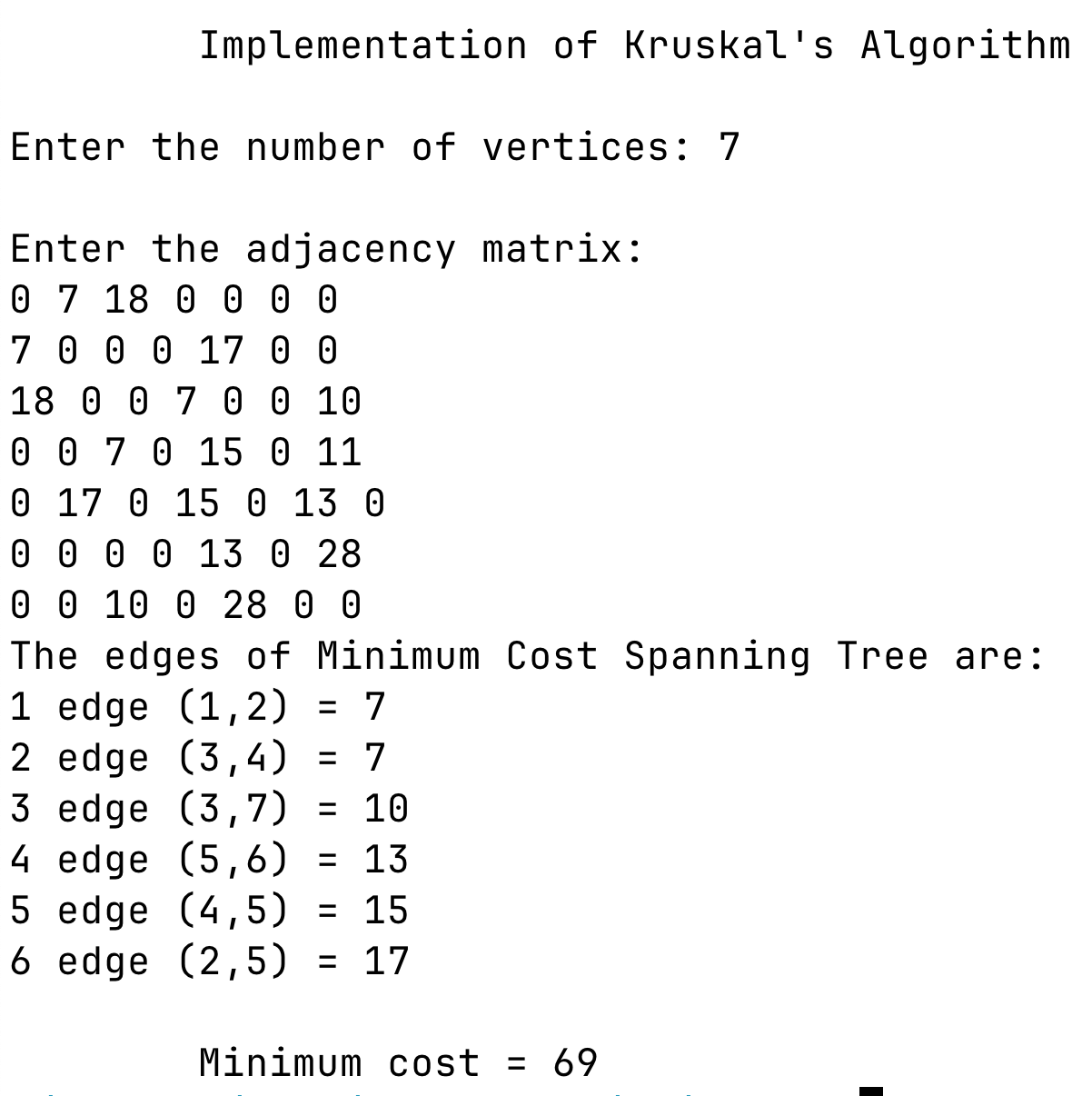
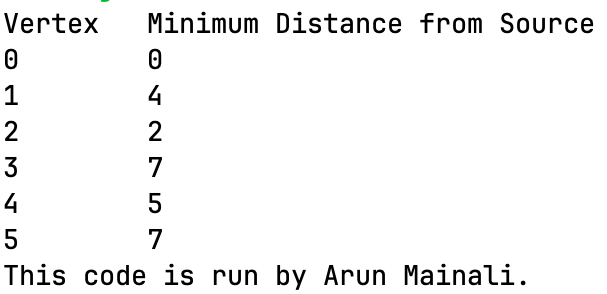
 

Figure 1: Output of Kuskal's algorithm

Figure 1: Output of Dijkstra's algorithm